**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

Ảnh có chứa biểu tượng, Nhãn hiệu, Phông chữ, vòng tròn

Mô tả được tạo tự động

**BÀI TẬP LỚN**

**MÔN: CÔNG NGHỆ JAVA**

**ĐỀ TÀI: LẬP TRÌNH GAME NINJA SCHOOL BẰNG JAVA**

**GIẢNG VIÊN HƯỚNG DẪN: Đào Thị Lệ Thủy**

**Thành viên nhóm:**

|  |  |  |
| --- | --- | --- |
| **Họ Và Tên:**  **Mai Đức Vinh**  **Nguyễn Mạnh Hùng**  **Trịnh Mạnh Quang** | **Lớp:**  **CNTT4 K63**  **CNTT4-K63**  **CNTT4-K63** | **Mã Sinh Viên**  **221231051**  **221220848**  **221230965** |

Hà Nội, năm 2024

**TRƯỜNG ĐẠI HỌC GIAO THÔNG VẬN TẢI HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

Ảnh có chứa biểu tượng, Nhãn hiệu, Phông chữ, vòng tròn

Mô tả được tạo tự động

**BÀI TẬP LỚN**

**MÔN: CÔNG NGHỆ JAVA**

**ĐỀ TÀI: LẬP TRÌNH GAME NINJA SCHOOL BẰNG JAVA**

**GIẢNG VIÊN HƯỚNG DẪN: Đào Thị Lệ Thủy**

**Thành viên nhóm:**

|  |  |  |
| --- | --- | --- |
| **Họ Và Tên:**  **Mai Đức Vinh**  **Nguyễn Mạnh Hùng**  **Trịnh Mạnh Quang** | **Lớp:**  **CNTT4 K63**  **CNTT4-K63**  **CNTT4-K63** | **Mã Sinh Viên**  **221231051**  **221220848**  **221230965** |

Hà Nội, năm 2024

# LỜI CẢM ƠN

Trong hành trình học tập và phát triển, sự hướng dẫn của một người thầy tận tâm là điều không thể thiếu. Cô Đào Thị Lệ Thủy không chỉ là người truyền đạt kiến thức, mà còn là nguồn cảm hứng, là ngọn đuốc soi đường cho những tâm hồn đang tìm kiếm sự hiểu biết. Sự kiên nhẫn và lòng nhiệt huyết của cô đã tạo nên một môi trường học tập lý tưởng, nơi mỗi câu hỏi đều được đón nhận, mỗi ý kiến đều được trân trọng. Qua từng bài giảng, cô đã không chỉ truyền đạt kiến thức về Java mà còn giáo dục chúng em về tinh thần trách nhiệm, sự tự lập và khả năng tư duy phản biện. Những bài học đó không chỉ giới hạn trong lớp học mà còn vang vọng lâu dài trong tâm trí mỗi học viên, trở thành kim chỉ nam cho những quyết định và hành động trong tương lai. Cô đã không chỉ là giảng viên, mà còn là người cố vấn đáng tin cậy, người bạn đồng hành trong mỗi dự án, mỗi thử thách mà chúng em phải đối mặt. Sự hỗ trợ vững chắc của cô đã giúp chúng em vượt qua những khó khăn ban đầu, khám phá ra khả năng tiềm ẩn và phát triển những kỹ năng quan trọng. Cô đã dạy chúng em cách để không chỉ viết code, mà còn để hiểu code, để cảm nhận và tận dụng triệt để sức mạnh của ngôn ngữ lập trình. Nhờ có cô, chúng em đã có thể nhìn thấy được những cơ hội mà công nghệ mang lại, và cách chúng ta có thể áp dụng nó để tạo ra những giải pháp sáng tạo cho thế giới xung quanh. Với lòng biết ơn sâu sắc, chúng em xin kính chúc cô sức khỏe, hạnh phúc và thành công trong mọi lĩnh vực. Cảm ơn cô đã là ngọn hải đăng dẫn lối, là nguồn động viên không ngừng nghỉ, và là tấm gương sáng cho chúng em noi theo. Chúng em tự hào khi được gọi cô là giảng viên của mình và mong rằng, dù thời gian có trôi qua, dù chúng em có đi đến bất cứ đâu, chúng em sẽ luôn ghi nhớ và trân trọng những giá trị mà cô đã truyền đạt. Cảm ơn cô, từ tận đáy lòng.

# LỜI NÓI ĐẦU

Trong khuôn khổ của môn học này, chúng em đã có cơ hội nghiên cứu và áp dụng các kiến thức cơ bản và nâng cao về lập trình Java vào việc thực hiện bài tập lớn. Đây là một cơ hội quý báu để chúng em rèn luyện kỹ năng lập trình, hiểu sâu hơn về ngôn ngữ Java, và trau dồi khả năng làm việc nhóm.

Báo cáo này sẽ giới thiệu về đề tài mà nhóm chúng em đã lựa chọn, mô tả về vấn đề được giải quyết, cũng như các phương pháp và công nghệ đã áp dụng trong quá trình thực hiện. Chúng em cũng sẽ chia sẻ những kết quả đạt được, những khó khăn đã gặp phải, và những bài học quan trọng từ quá trình này.

Trong báo cáo này, chúng em sẽ trình bày về quá trình phát triển và các thành phần chính của trò chơi RPG mà chúng em đã tạo. Đây là một dự án quan trọng cho chúng em, nơi chúng em đã áp dụng những kiến ​​thức về lập trình hướng đối tượng và Java để tạo ra một trải nghiệm game thú vị.

Trong quá trình phát triển trò chơi này, chúng em đã đối mặt với nhiều thách thức và học được rất nhiều bài học quý giá về quản lý dự án, lập trình Java và tương tác người dùng. Chúng em rất mong muốn có cơ hội chia sẻ những kết quả và kinh nghiệm này với quý cô và các bạn.

# MỤC LỤC

[LỜI CẢM ƠN 3](#_Toc164640149)

[LỜI NÓI ĐẦU 4](#_Toc164640150)

[MỤC LỤC 5](#_Toc164640151)

[CHƯƠNG 1: ĐẶT VẤN ĐỀ 6](#_Toc164640152)

[CHƯƠNG 2: HƯỚNG DẪN CHƠI 8](#_Toc164640153)

[2.1 Nhân vật, quái vật và bản đồ 8](#_Toc164640154)

[2.1.1 Thiết lập nhân vật và bản đồ 8](#_Toc164640155)

[2.1.2 Thiết lập nhân vật và quái 8](#_Toc164640156)

[2.1.3 Di chuyển nhân vật, quái vật 9](#_Toc164640157)

[2.2 Menu Game, Pause Game 9](#_Toc164640158)

[2.2.1 Menu Game 9](#_Toc164640159)

[2.2.2 Tạm dừng trò chơi 11](#_Toc164640160)

[2.3 Cơ chế Kết thúc 11](#_Toc164640161)

[CHƯƠNG 3: THIẾT KẾ CÁC ĐỐI TƯỢNG 13](#_Toc164640162)

[3.1 Giới thiệu về chương trình 13](#_Toc164640163)

[3.2 Cấu trúc chương trình 13](#_Toc164640164)

[3.2.1 Package controller 13](#_Toc164640165)

[3.2.2 Package model 13](#_Toc164640166)

[3.2.3 Package view 14](#_Toc164640167)

[CHƯƠNG 4: KẾT QUẢ ĐẠT ĐƯỢC 16](#_Toc164640168)

[4.1 Thiết lập cơ chế vòng lặp cho trò chơi 16](#_Toc164640169)

[4.2 Thiết lập nhân vật 18](#_Toc164640170)

[4.2.1 Cơ chế di chuyển của nhân vật 18](#_Toc164640171)

[4.2.2 Cơ chế nhân vật nhảy 18](#_Toc164640172)

[4.2.3 Hiệu ứng tấn công của nhân vật và cơ chế tấn công quái vật của nhân vật 19](#_Toc164640173)

[4.2.4 Thiết lập cơ chế va chạm của người chơi với từng khối của bản đồ 20](#_Toc164640174)

[4.2.5 Thông số nhân vật và nâng cấp 22](#_Toc164640175)

[4.2.6 Tấn công chiêu thức 22](#_Toc164640176)

[4.3 Thiết lập bản đồ 22](#_Toc164640177)

[4.3.1 Thiết lập bản đồ với cơ chế tải từng khối hộp 23](#_Toc164640178)

[4.3.2 Cơ chế chuyển bản đồ 23](#_Toc164640179)

[4.3. Thiết lập đối với các quái vật có trong game 4.3.1. Máu và cách thức hoạt động 24](#_Toc164640180)

[4.3.2. Khả năng phát hiện người chơi và hiệu ứng tấn công của quái 25](#_Toc164640181)

[4.3.3. Khả năng tự sinh lại quái sau khi bị tiêu diệt 26](#_Toc164640182)

[4.4. Thiết lập đối với các thực thể trong game 27](#_Toc164640183)

[4.4.1. Thuốc hồi máu và năng lượng 27](#_Toc164640184)

[4.4.2. Hệ thống hộp 28](#_Toc164640185)

[4.4.3 Súng thần công 29](#_Toc164640186)

[4.4.4. Cạm bẫy 31](#_Toc164640187)

[CHƯƠNG 5: CẢI THIỆN VÀ NÂNG CẤP 32](#_Toc164640188)

[5.1. Vấn đề cơ sở dữ liệu 32](#_Toc164640189)

[5.2. Một số tính năng có thể phát triển thêm 32](#_Toc164640190)

[5.3. Cải thiện về quái 32](#_Toc164640191)

[PHÂN CHIA NHIỆM VỤ 33](#_Toc164640192)

[TÀI LIỆU THAM KHẢO 34](#_Toc164640193)

# CHƯƠNG 1: ĐẶT VẤN ĐỀ

Trò chơi nhập vai, hay RPG, là một phần không thể thiếu trong ngành công nghiệp game hiện đại, nơi mà người chơi có thể đắm chìm trong những câu chuyện hấp dẫn và thể hiện bản thân qua các nhân vật anh hùng. Sự phát triển của trò chơi Ninja School là một bước tiến tự nhiên trong bối cảnh này, nhất là khi xét đến sức hút không ngừng của văn hóa ninja và anime/manga Nhật Bản. Đây không chỉ là một trò chơi giải trí mà còn là cơ hội để người chơi tìm hiểu sâu hơn về văn hóa phong phú này.

Ninja School hứa hẹn sẽ mang lại một không gian sống động, nơi người chơi có thể khám phá và tương tác với thế giới ninja một cách chân thực. Từ việc học các kỹ năng chiến đấu, thực hiện nhiệm vụ, đến việc tìm hiểu về lịch sử và truyền thống của ninja, mỗi khía cạnh của trò chơi đều được thiết kế để tạo ra một trải nghiệm đắm chìm và giáo dục. Điều này không chỉ giúp người chơi cảm thấy thích thú với các hoạt động trong game mà còn giúp họ hiểu rõ hơn về một phần quan trọng của văn hóa Nhật Bản.

Sự kết hợp giữa lối chơi nhập vai truyền thống và yếu tố văn hóa đặc sắc này tạo nên một sản phẩm có sức hấp dẫn đặc biệt. Người chơi không chỉ được thử thách qua các màn chơi đa dạng mà còn được thưởng thức một câu chuyện phong phú, đầy màu sắc với những nhân vật và tình tiết được xây dựng công phu. Điều này giúp Ninja School không chỉ là một trò chơi mà còn là một tác phẩm nghệ thuật, nơi mà mỗi quyết định của người chơi đều có thể tạo ra những hướng đi mới mẻ cho câu chuyện.

Ngoài ra, việc tích hợp các yếu tố xã hội vào trò chơi cũng là một điểm nhấn quan trọng, cho phép người chơi không chỉ tương tác với thế giới ảo mà còn kết nối với cộng đồng game thủ khác. Điều này tạo điều kiện cho việc hình thành các mối quan hệ mới, cũng như việc chia sẻ kinh nghiệm và chiến lược giữa người chơi, làm phong phú thêm trải nghiệm chung.

Với sự chú trọng vào chi tiết và sự sáng tạo không ngừng, Ninja School có tiềm năng trở thành một trong những trò chơi RPG hàng đầu, không chỉ thu hút người hâm mộ của thể loại này mà còn thu hút những người yêu thích văn hóa Nhật Bản. Sự kết hợp giữa lối chơi hấp dẫn và nội dung sâu sắc hứa hẹn sẽ mang lại cho người chơi một trải nghiệm không thể quên. Đây chắc chắn sẽ là một bước tiến mới trong ngành công nghiệp game, nơi mà sự đổi mới luôn được đón nhận với niềm hứng khởi.

# CHƯƠNG 2: HƯỚNG DẪN CHƠI

## 2.1 Nhân vật, quái vật và bản đồ

### 2.1.1 Thiết lập nhân vật và bản đồ

Trò chơi sử dụng một bản đồ có kích thước 30x70. Nhân vật sẽ do một người điều khiển. Nhân vật ban đầu ở bản đồ 1 và ở một tọa độ cố định. Trên bản đồ sẽ có các chướng ngại vật và công trình để cho nhân vật di chuyển và tương tác với chúng. Với mỗi một bản đồ sẽ có một cổng không gian dịch chuyển đến bản đồ mới. Người chơi sẽ dựa vào khả năng của mình để di chuyển nhân vật sao cho hợp lý đến vượt qua các chướng ngại vật và công trình để đi đến vị trí mình mong muốn.

Ảnh có chứa ảnh chụp màn hình, ngôi nhà, cửa sổ

Mô tả được tạo tự động

Hình 2.1 Tổng quan trò chơi

### 2.1.2 Thiết lập nhân vật và quái

Nhân vật ban đầu sẽ ở cấp độ một với lượng máu mặc định và lượng sát thương mặc đinh. Người chơi sẽ đi tìm quái ở mỗi bản đồ đánh quái và tăng cấp độ của mình. Sau khi người chơi lên cấp độ mới lượng máu và lượng sát thương của người chới sẽ tăng lên. Với mỗi con quái và khu vực quái sẽ có lượng máu và lượng sát thương khác nhau. Người chơi phải thực hiện khéo léo di chuyển và tân công quái một cách hợp lý để giết chết quái vật và lên cấp.

Ảnh có chứa ảnh chụp màn hình, núi, bầu trời, điểm ảnh

Mô tả được tạo tự động

Hình 2.2 Tương tác nhân vật với quái vật

### 2.1.3 Di chuyển nhân vật, quái vật

Nhân vật sẽ di chuyển sang trái phải bằng hai nút lần lượt là A và D.

Nhân vật sẽ nhảy bằng nút W.

Nhân vật sẽ đánh bằng nút Space.

Nhân vật sẽ tung kỹ năng phi tiêu bằng nút H.

Quái vật sẽ di chuyển quanh các vị trí được chỉ định hoặc sẽ đứng yên với một số quái đặc biệt

## 2.2 Menu Game, Pause Game

### 2.2.1 Menu Game

Trong màn hình menu game có 3 nút chính là PLAY, OPTIONS và QUIT.

Ảnh có chứa văn bản, ảnh chụp màn hình, thiết kế đồ họa, Game PC

Mô tả được tạo tự động

Hình 2.3 Menu Game

Khi người chơi click vào PLAY màn hình chính của game sẽ chuyển sang màn hình chơi game và màn hình chơi game.

Khi người chơi click vào OPTIONS màn hình chính của game sẽ hiển thị thêm một giao diện option để người chơi điều chỉnh ầm thanh của game.

Ảnh có chứa văn bản, thiết kế đồ họa, ảnh chụp màn hình, phim hoạt hình

Mô tả được tạo tự động

Hình 2.4 Giao diện cài đặt của trò chơi

Khi người chơi click vào QUIT game sẽ được tắt và giao diện game sẽ đóng lại.

### 2.2.2 Tạm dừng trò chơi

Khi người chơi đang ở trong màn hình của trò chơi người dùng có thể ấn nút P để thực hiện dừng trò chơi lại và hiện ra giao diện tạm dừng.

Ảnh có chứa mây, văn bản, bầu trời, ảnh chụp màn hình

Mô tả được tạo tự động

Hình 2.5 Giao diện tạm dừng của trò chơi

Trong màn hình tạm dừng của trò chơi chúng ra có thể điều chỉnh âm thanh như tùy chỉnh và ngoài ra còn có một số nút khác như là trở về màn hình menu nút trở về màn 1 của game và nút hủy bỏ trạng thái dừng của trò chơi trở lại màn hình trò chơi trước lúc dừng.

## 2.3 Cơ chế Kết thúc

Khi người chơi mất máu đến lúc lượng máu của người chơi nhỏ hơn 0 thì lúc đó người chơi sẽ chết và hiển thị giao diện khi thua trò chơi.

Ảnh có chứa ảnh chụp màn hình, phim hoạt hình, văn bản

Mô tả được tạo tự động

Hình 2.6 Giao diện thua cuộc

Khi đó bạn sẽ có 2 lựa chọn là trở lại giao diện chính hoặc trở lại màn 1.

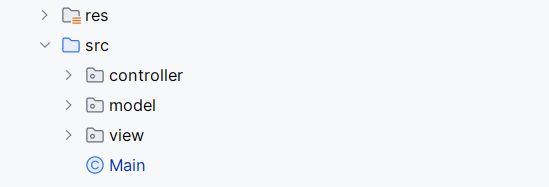
# CHƯƠNG 3: THIẾT KẾ CÁC ĐỐI TƯỢNG

## 3.1 Giới thiệu về chương trình

Trò chơi được thiết kế dựa trên mô hình MVC.

* Model (Lưu trữ toàn bộ cấu trúc dữ liệu, logic game)
* View (Chịu trách nhiệm xử lý giao diện)
* Controller **(**Lấy tương tác người dùng để lấy dữ liệu của **model** rồi gửi vào **view** để nhận lại giao diện)

Cấu trúc chương trình:



Hình 3.1 Cấu trúc chương trình

## 3.2 Cấu trúc chương trình

Lưu trữ hình ảnh và âm thanh được sử dụng trong trò chơi trong thư mục res.

Class **Main**: Khởi chạy trò chơi.

### 3.2.1 Package controller

Class **KeyboardInputs**: Lấy tương tác người dùng từ bàn phím để lấy dữ liệu của **model** rồi gửi vào **view** để nhận lại giao diện. Cập nhật dữ liệu của trò chơi, điều khiển trò chơi.

Class **MouseInputs**: Lấy tương tác người dùng từ bàn phím để lấy dữ liệu của **model** rồi gửi vào **view** để nhận lại giao diện. Cập nhật dữ liệu của trò chơi, điều khiển trò chơi.

### 3.2.2 Package model

Package **entities**:

* Abstract class **Entity**: Lưu trữ thông tin chung của các thực thể trong trò chơi
* Class **Player**: Lưu trữ thông tin của nhân vật
* Class **Enemy**: Lưu trữ thông tin chung của quái vật
* Class **Crab**: Lưu trữ thông tin về quái vật cua
* Class **Dummy** (extends **Enemy**): Lưu trữ thông tin về quái vật bù nhìn.
* Class **Reaper** (extends **Enemy**): Lưu trữ thông tin về quái vật ma.
* Class **Samurai** (extends **Enemy**): Lưu trữ thông tin về quái vật tướng.
* Class **Tengu** (extends **Enemy**): Lưu trữ thông tin về quái vật thiên cẩu.
* Class **EnemyManager**: Khởi tạo và điều khiển trạng thái của quái vật.

Package **gamestates**:

* Interface **Statemethod**:
* Class **GameOptions**: Quản lý giao diện và trạng thái trò chơi.
* Enum **Gamestate**: Lưu các trạng thái trò chơi.
* Class **Playing**: Lưu trữ và xử lý trạng thái hoạt động trò chơi.
* Class **State**: Điều khiển trạng thái trò chơi.

Package **maps**:

* Class **Map**: Lưu trữ thông tin của bản đồ.
* Class **MapManager**: Xử lý sự kiện đổi map của trò chơi.

Package **objects**:

* Class **GameObject**: Lưu trữ thông tin chung của các đồ vật trong trò chơi.
* Class **ObjectManager**: Khởi tạo và quản lý các đồ vật.
* Class **GameContainer** (extends **GameObject**): Xử lý sự kiện của đồ vật.
* Class **Cannon** (extends **GameObject**): Lưu trữ thông tin của khẩu pháo.
* Class **Potion** (extends **GameObject**): Lưu trữ thông tin của bình thuốc.
* Class **Spike** (extends **GameObject**): Lưu trữ thông tin của bẫy.
* Class **Projectile**: Lưu trữ thông tin của đạn.

Package **utilz**:

* Class **Constants**: Lưu trữ các hằng số của trò chơi.
* Class **HelpMothods**: Lưu trữ các phương thức để xử lý sự kiện trò chơi
* Class **LoadSave**: Lưu trữ ảnh và bản đồ của trò chơi

Class **AudioPlayer**: Lưu trữ âm thanh của nhân vật

Class **Game**: Lữu trữ thông số trò chơi

### 3.2.3 Package view

Class **GamePanel**: Xử lý giao diện trò chơi.

Class **GameWindow**: Xử lý giao diện của khung cửa sổ trò chơi.

Class **AudioOptions**: Xử lý giao diện âm thanh.

Class **GameOverOverlay**: Xử lý hình ảnh nhân vật khi thua.

Class **MenuButton**: Xử lý giao diện vào trò chơi.

Class **PauseButton**: Xử lý nút dừng trò chơi.

Class **PauseOverlay**: Xử lý trạng thái dừng trò chơi.

Class **SoundButton** (extends **PauseButton**): Xử lý nút chỉnh âm thanh.

Class **UrmButton** (extends **PauseButton**): Xử lý nút về trang chủ.

Class **VolumeButton** (extends **PauseButton**): Xử lý nút chỉnh âm lượng.

Class **Background**: Xử lý hình nền trò chơi.

# CHƯƠNG 4: KẾT QUẢ ĐẠT ĐƯỢC

## 4.1 Thiết lập cơ chế vòng lặp cho trò chơi

Trong một trò chơi với cơ chế hoạt động với những hoạt ảnh như trò chơi Ninja School chúng ta cần phải thiết lập cơ chế vòng lặp liên tục cho trò chơi nghĩa là mỗi một khoảng thời gian nhất định trò chơi sẽ được cập nhật hoạt ảnh theo thời gian để người chơi có thể nhận thấy sự tương tác của bản thân với nhân vật và cũng như giúp trò chơi được lôi cuốn với người chơi hơn.

Cơ chế này được thiết lập bằng cách khi khởi động trò chơi một lớp thread sẽ được tạo ra với chính đối tượng là trò chơi và các phương thức khởi chạy trong hàm run() của trò chơi sẽ được chạy và tạo ra một vòng lặp.

Ảnh có chứa tác phẩm nghệ thuật, Hoạt hình, bức vẽ, phim hoạt hình

Mô tả được tạo tự động

Hình 4.1 Hinh ảnh minh họa về vòng lặp trò chơi

Với trò chơi này ta phải tạo ra một vòng lặp ổn định vì thế cơ chế vòng lặp này sẽ đưa ra các thiết lập củ thể về số khung hình và số cập nhật cho trò chơi như sau:

Ban đầu ta sẽ tạo ra hai biến là số khung hình cập nhật trong 1 giây và số cập nhật cập nhật trong 1 giây. Sau đó ta tạo ra biến đếm số khung hình và số cập nhật được khởi tạo với giá trị ban đầu là 0. Tiếp theo ta tạo biến lưu thời gian ban đầu trước mỗi vòng lặp và thời gian cuối cùng để tính toán thời gian giữa các vòng lặp. Và ta tạo ra hai biến để theo dõi thời gian đã trôi qua trong cho mỗi khung hình và cập nhật được khởi tạo với giá trị ban đầu bằng 0.

Tiếp đến sẽ là một vòng lặp vô hạn để chạy trò chơi. Trong vòng lặp này sẽ thực hiện các nội dung lần lượt. Đầu tiên sẽ lấy thời gian hiện tại sau đó tính toán thời gian cho khung hình và cập nhật và lưu vào biến theo dõi thời gian ứng với mỗi phần. Cách tính biến theo dõi thời gian khung hình và cập nhật được tính bằng công thức biến theo dõi thời gian hiện tạo cộng với thời gian hiện tại trừ thời gian ban đầu tất cả chia cho số khung hình trong 1 giây hoặc số cập nhật trong 1 giây đối với thời gian theo dõi cập nhật. Nếu hai giá trị thời gian này có giá trị nào lớn hơn hoặc bằng 1 thì ta sẽ gọi phương thức vẽ lại khung hình đỗi với khung hình hoặc gọi lại phương thức cập nhật lại trạng thái trò chơi đối với cập nhật sau đó ta tăng số lượng khung hình hoặc số lượng cập nhật cuối cùng ta giảm biến theo dõi thời gian. Và cuối cùng trong vòng lặp này ta kiểm tra xem đã đến 1 giây chưa bằng cách kiểm tra xem thời gian cuối cùng của vòng lặp hiện tại trừ đi thời gian cuối cùng ban đầu ta khai báo để tính toán nếu giá trị tính được lớn hơn 1000 thì ta sẽ cập nhật lại giá trị cuối cùng và đặt lại biến đếm số lượng khung hình và số lượng cập nhật bằng 0.

Vòng lặp của trò chơi sẽ được tiếp diễn đến khi chúng ta tắt trò chơi bởi vì đó là một vòng lặp vô hạn khi chương trình được khởi động.

|  |
| --- |
| private void startGameLoop() {  gameThread = new Thread(this);  gameThread.start(); }  public void run() {  double timePerFime = 1000000000.0 / FPS\_SET;  double timePerUpdate = 1000000000.0 / UPS\_SET;  long previousTime = System.*nanoTime*();  int frames = 0;  int updates = 0;  long lastCheck = System.*currentTimeMillis*();  double deltaU = 0, deltaF = 0;  while (true) {  long currentTime = System.*nanoTime*();  deltaU += (currentTime - previousTime) / timePerUpdate;  deltaF += (currentTime - previousTime) / timePerFime;  previousTime = currentTime;  if (deltaU >= 1) {  update();  updates++;  deltaU--;  }  if (deltaF >= 1) {  gamePanel.repaint();  deltaF--;  frames++;  }  if (System.*currentTimeMillis*() - lastCheck >= 1000) {  lastCheck = System.*currentTimeMillis*();  frames = 0;  updates = 0;  }  } } |

## 4.2 Thiết lập nhân vật

Khi chúng ta sử dụng các phím để điều khiển nhân vật thì phương thức ứng với phím đấy sẽ trả về giá trị đúng và trong lớp nhân vật sẽ thực hiện phương thức đó. Sau đó khi ta thả phím để điều khiển ra phương thức ứng với phím đó sẽ trả về giá trị sai khi ấy trong lớp nhân vật sẽ không thực hiện phương thức mình vừa điều khiển nữa mà sẽ đưa nhân vật về trạng thái bình thường.

### 4.2.1 Cơ chế di chuyển của nhân vật

Khi người chơi sử dụng các phím A hoặc D thì phương thức di chuyển trái phải của nhân vật sẽ đưa về giá trị đúng và lúc đó nhân vật sẽ di chuyển với cơ chế. Khi người chơi ấn phím A thì tọa độ x của nhân vật sẽ được giảm với giá trị bằng tốc độ của nhân vật và củ thệ trong trò chơi giá trị ấy bằng 2 còn khi người chơi ấn phím D tọa độ x của nhân vật sẽ được tăng lên 2.

Và đồng thời khi nhân vật di chuyển cũng sẽ thực hiện hoạt ảnh tương ứng với mỗi tương tác của người chơi.

Ảnh có chứa phim hoạt hình, con chim, ngoài trời

Mô tả được tạo tự độngẢnh có chứa phim hoạt hình, ảnh chụp màn hình, tác phẩm nghệ thuật

Mô tả được tạo tự động

Hình 4.2 Hình ảnh chuyển động của nhân vật

### 4.2.2 Cơ chế nhân vật nhảy

Khi người chơi ấn vào nút W phương thức nhảy của nhân vật sẽ trả về giá trị đúng. Lúc đó tốc độ bay của nhân vật sẽ được gán bằng -2.25 lúc đó tọa độ y của nhân vật sẽ được giảm nhanh dần khiến cho người ta có cảm giác nhân vật bay lên đồng thời tốc độ bay của nhân vật sẽ được tăng dần với giá trị trọng lực là 0.04 để cho nhân vật đến một mức nhất định tốc độ bay sẽ bằng 0 và khi nhân vật có tốc độ bay lớn hơn 0 thì tọa độ y của nhân vật sẽ được tăng dần và cho người ta có cảm giác nhân vật đang rơi dần xuống.

Ảnh có chứa phim hoạt hình, ảnh chụp màn hình, hoạt hình Nhật Bản

Mô tả được tạo tự động

Hình 4.3 Hình ảnh nhảy của nhân vật

Và đồng thời nhân vật sẽ thực hiện hoạt ảnh tưng ứng với ba mức độ nhảy đó là lúc bắt đầu nhảy lúc đang trên không một khoảng thời gian và lúc rơi xuống.

Ảnh có chứa bản phác thảo, phim hoạt hình, Hoạt hình, hình mẫu

Mô tả được tạo tự độngẢnh có chứa vòng tròn, màu xanh lam, Xanh điện

Mô tả được tạo tự độngẢnh có chứa bản phác thảo, phim hoạt hình, Hoạt hình, Phim hoạt hình

Mô tả được tạo tự động

Hình 4.4 Hoạt ảnh nhảy của nhân vật

### 4.2.3 Hiệu ứng tấn công của nhân vật và cơ chế tấn công quái vật của nhân vật

Khi người chơi nhấn nút Space phương thức tấn công của nhân vât sẽ trả về giá trị đúng. Và sau đó hoạt ảnh tấn công của nhân vật được thực hiện nhưng khi đang thực hiện hoạt ảnh tấn công người chơi phải đợi cho đến khi nhân vật tấn công xong mới được tấn công tiếp để tránh trường hợp người chơi dùng đòn đánh liên tục bằng cách dùng một biến được tạo ra để kiểm tra xem nhân vật đang ở hoạt ảnh cuối cùng hay chưa nếu đã ở hoạt ảnh cuối cùng thì biến kiểm tra đó sẽ đưa về giá trị sai nhằm biểu thì rằng nhân vật đã không còn trạng thái đang tân công nữa và cho phép người chơi tấn công tiếp.



Hình 4.5 Hoạt ảnh tấn công của nhân vật

Nhân vật sẽ được cung cấp một khoảng tấn công vừa đủ so với hình ảnh của nhân vật, khi nhân vật phát động tấn công nếu quái vật nằm trong khu vực tấn công của nhân vật thì quái vật sẽ phải nhận đúng với lượng sát thương hiện tại của nhân vật.

Ảnh có chứa ảnh chụp màn hình, bầu trời

Mô tả được tạo tự động

Hình 4.6 Khoảng cách tấn công của nhân vật

### 4.2.4 Thiết lập cơ chế va chạm của người chơi với từng khối của bản đồ

Nhân vật được thiết lập hitbox quanh hoạt ảnh của nhân vật. Mỗi một cạnh của nhân vật sẽ được tương tác với mỗi cạnh của 1 khối của bản đồ.

Đối với các khối không di chuyển được, và không di chuyển được:

Bằng cách dùng phương thức kiểm tra lấy tọa độ 4 góc của hitbox nhân vật. Đầu tiên ta kiểm tra xem nó có thuộc vào các khối được xét là đi qua được hay không nếu nó không thuốc vào khối đi qua được thì phương thức kiểm tra các góc sẽ trả về giá trị đúng. Rồi đến hàm kiểm tra xem nhân vật có đi qua được không sẽ trả về giá trị sai. Còn nếu là khối đi qua được thì sẽ trả về giá trị đúng từ đó ta sẽ cập nhật được vị trí của hitbox nhân vật đối với các khối.

Ảnh có chứa bầu trời, ảnh chụp màn hình, phim hoạt hình

Mô tả được tạo tự động

Hình 4.7 Hình ảnh va chạm của nhân vật

Đối với các khối đặc biệt ví dụ như mái và cầu:

Tương tự với phương thức kiểm tra va chạm nhưng ta sẽ mặc định nó ban đầu là khối di chuyển được trái phải tức là cập nhật được tọa độ x trong khối đó. Nhưng ngoài ra ta sẽ sử dụng phương thức kiểm tra va chạm chỉ cập nhật vị trí 2 góc dưới để kiểm tra. Kiểm tra với các điều kiện là khi nhân vật ở trên không trung lúc đang ở trạng thái nhảy lên (lúc mà tốc độ bay nhỏ 0) chưa rơi xuống phương thức kiểm tra va chạm sẽ đưa ra kết quả là đúng tức là lúc đó nhân vật sẽ nhảy qua khối đó và tọa độ hitbox y sẽ được cập nhật liên tục. Khi nhân vật ở trạng thái rơi xuống (lúc mà tốc độ bay lớn hơn 0) lúc ấy sẽ dùng phương thức kiểm tra xem khối đấy có phải khối đặc biệt không nếu là khối đặc biệt thì nhân vật sẽ chạm vào khối đấy được còn nếu không thì nhân vật sẽ tiếp tục rơi xuống.

Ảnh có chứa bức vẽ, phim hoạt hình, văn bản, minh họa

Mô tả được tạo tự động

Hình 4.8 Hình ảnh va chạm trên mái

### 4.2.5 Thông số nhân vật và nâng cấp

Ban đầu nhân vật sẽ được trao cho giá trị sức mạnh cơ bản để dễ dàng hoạt động trong thế giới nguy hiểm nhưng khi lên cấp nhân vật sẽ được tăng giá trị sức mạnh tương ứng với cấp độ đó.

Ảnh có chứa ảnh chụp màn hình, Nhiều màu sắc, hàng, Hình chữ nhật

Mô tả được tạo tự động

Hình 4.9 Thanh trạng thái của nhân vật

Khi nhân vật tiêu diệt quái vật sẽ được tăng một lượng điểm kinh nghiệm tương ứng với lượng sức mạnh của quái. Khi thanh kinh nghiệm hiện tại của nhân vật đạt đủ với thanh kinh nghiệm tối đa thì nhân vật sẽ được nâng cấp.

Khi nhân vật tăng cấp sẽ được hồi lại đầy thanh máu và thanh năng lượng, và thanh kinh nghiệm của nhân vật sẽ về 0. Khi đó lượng sức mạnh của nhân vật sẽ tăng lên tương ứng với cấp độ hiện tại của nhân vật

Ảnh có chứa văn bản, phim hoạt hình, ảnh chụp màn hình, biển hiệu

Mô tả được tạo tự động

Hình 4.10 Trạng thái nâng cấp của nhân vật

### 4.2.6 Tấn công chiêu thức

Nhân vật sẽ có một điểm năng lượng để dùng cho việc tấn công bằng chiều thức gây ra lượng sát thương lớn lên quái vật

Ảnh có chứa phim hoạt hình, cỏ, ngoài trời, tác phẩm nghệ thuật

Mô tả được tạo tự động

Hình 4.11 Hình ảnh sử dụng tuyệt chiêu

## 4.3 Thiết lập bản đồ

Trong một trò chơi bản đồ là một thứ không thể thiếu bởi vì đó chính là thứ thu hút người chơi giúp người chơi tương tác với trò chơi tốt hơn. Trong trò chơi này sẽ có 3 loại bản đồ khác nhau.

### 4.3.1 Thiết lập bản đồ với cơ chế tải từng khối hộp

Mỗi một khối hộp trên bản đồ sẽ ứng với kích thước 32x 32px. Mỗi một khối hộp ấy sẽ được ứng với một thành phần đặc biệt của bản đồ.

Cơ chế thiết lập bản đồ chính là lấy các khối block từ trong ảnh để chuyển đổi các ô trong ảnh thành các sô để cho tiện lấy các khối tương ứng với các số bởi vì ở đây chúng ta vẽ bản đồ bằng mảng số 2 chiều.

Khi chúng ta lấy mảng 2 chiều trong phương thức tải ảnh của trò chơi trong đó trò chơi sẽ lấy các khối tương ứng với các số và tải bức ảnh của số đầy tương ứng với vị trí của giao diện trò chơi từ đó chúng ta tải được bản đồ hoàn chỉnh

Ảnh có chứa hàng, Song song

Mô tả được tạo tự động

Hình 4.12 Hình ảnh bản đồ của trò chơi

### 4.3.2 Cơ chế chuyển bản đồ

Bởi vì để cho trò chơi được thú vị nên ta đã tạo ra 3 bản đồ với mỗi bản đồ sẽ có một cổng dịch chuyển sang bản đồ khác. Cơ chế chuyển bản đồ sẽ được thực hiện khi chúng ta di chuyển vào cổng dịch chuyển bản đồ người chơi quái vật và các thành phần hỗ trợ sẽ được tải lại ứng với mỗi bản đồ. Khi chúng ta chuyển bản đồ ta có một biến để lưu xem chúng ta đang muốn chuyển đến bản đồ nào và cơ chế cập nhật sẽ cập nhật được những thứ xung quanh bản đồ đó từ ấy bản đồ chúng ta sẽ được tạo mới.

Cách thiết lập cổng dịch chuyển như sau. Đầu tiên thiết lập cổng dịch chuyển là một khối có thể đi qua được sau đó ta chuyển khối ấy ứng với phương thức kiểm tra xem chúng ta có đang ở trong khối cổng dịch chuyển chúng ta thiết lập hay không. Nếu chúng ta đang ở trong cổng dịch chuyển phương thức ấy sẽ trả về giá trị đúng. Cùng lúc ấy biến đổi cổng dịch chuyển sẽ được tăng lên 1 đơn vị và cập nhật sang bản đồ tiếp theo.



Hình 4.13 Hình ảnh cổng dịch chuyển

## 4.3. Thiết lập đối với các quái vật có trong game **4.3.1. Máu và cách thức hoạt động**

Các quái trong hệ thống sẽ đều được trang bị cho một lượng máu cố định và sẽ xuất hiện tại một vị trí chỗ định ở trong bản đồ của trò chơi.

Đối với mỗi quái đều sẽ được quy định cho một khu vực hoạt động riêng và quái sẽ được di chuyển đi lại trong khu vực đó.

Mỗi quái sẽ có một cách hoạt động khác nhau như “bù nhìn” thì chỉ đứng yên một chỗ, còn cách loại cách sẽ được di chuyển. Có 2 cách di chuyển cho quái một là di chuyển sang vài bước rồi quay đầu lại, hai là di chuyển qua lại khi thấy vật cản thì quay đầu.





Hình 4.14 Hình ảnh quái vật trong trò chơi

### 4.3.2. Khả năng phát hiện người chơi và hiệu ứng tấn công của quái

Mỗi quái sẽ được trao cho khả nắng phát hiện người chơi ở trong một khoảng cách nhất định. Nếu nhân vật di chuyên vào tầm nhìn của quái vật thì quái sẽ tìm đến chỗ nhân vật để tiêu diệt.

|  |
| --- |
| protected boolean canSeePlayer(int[][] lvData, Player player) {if (isPlayerInRange(player)) {  if (*isSightClear*(lvData, hitBox, player.hitBox))  return true;  }  return false;  } |

|  |
| --- |
| private boolean isPlayerInRange(Player player) {  float distanceX = Math.*abs*((int) (player.getHitBox().getCenterX() - this.getHitBox().getCenterX())-10);  float distanceY = Math.*abs*((int)(player.getHitBox().getCenterY() - this.getHitBox().getCenterY())-10);  float distance = (float) Math.*sqrt*(distanceX \* distanceX + distanceY \* distanceY);  return distance <= attackDistance\*1.5; } |

Mỗi quái sẽ được trang bị một hitbox riêng biệt tùy theo kích thước và mỗi khi hitbox đó va chạm hay có tương tác với hitbox của người chơi thì ngay lập tức hệ thống quái sẽ được chuyển sang trạng thái tấn công và lượng máu của người chơi sẽ bị trừ đi theo số sát thương được gán cho từng quái. Và sau khi mỗi con quái được tiêu diệt thì lượng kinh nghiệm nhất định sẽ được cộng cho người chơi.



Hình 4.15 Hình ảnh quái vật tìm đường đi của quái vật

### 4.3.3. Khả năng tự sinh lại quái sau khi bị tiêu diệt

Các loại quái vật sẽ có khả năng sinh sản lại sau khi bị người chơi tiêu diệt nhưng tính năng này mới chỉ được xuất hiện trên quái vật “Ma”.

|  |
| --- |
| public void update(int[][] lvData, Player player) {  ArrayList<Reaper> temp = new ArrayList<>();  int n = reapers.size();  for (Reaper X : reapers) {  if(X.alive) X.update(lvData, player);  else {  temp.add(X);  spawnMonster = true;  }  }  reapers.removeAll(temp);  if (frameCount <= DISPLAY\_FRAMES){  if (spawnMonster && n == 0 && frameCount == DISPLAY\_FRAMES) {  addReaperMap2();  spawnMonster = false;  }  frameCount++;  }else frameCount = 0;  } |



Hình 4.16 Hình ảnh quái vật Ma

## 4.4. Thiết lập đối với các thực thể trong game

### 4.4.1. Thuốc hồi máu và năng lượng

Đây là hai thực thể xuất hiện cố định trên bản đồ và luôn có chuyển động xoay quanh tại chỗ và khi hitbox của người chơi tương tác với hitbox thực thể thì lượng máu hoặc năng lượng cố định sẽ được cộng thêm vào cho người chơi.

Hình 4.17 Hình ảnh vật phẩm hỗ trợ trong trò chơi

### 4.4.2. Hệ thống hộp

Hệ thống hộp là thực thể dùng để chứa máu và năng lượng, khi người chơi tương tác với hai vật thể này thì năng lượng hoặc máu tương ứng sẽ được rơi ra theo từng hộp.



Hình 4.18 Hình ảnh hộp chứa vật phẩm

Tại class Player trong phương thức checkattack() gọi đến phương thức checkObjecthit nằm trong class Objectmanager. Phương thức này có tác dụng kiểm tra xem tấn công của nhân vật có tác động được đến hòm chứa hay không.

|  |
| --- |
| public void checkObjectHit(Rectangle2D.Float attackBox){  for(GameContainer gc : containers){  if(gc.isActive()){  if(gc.getHitbox().intersects(attackBox)){  gc.setAnimation(true);  int type = 21;  if(gc.getObjType() == *BARREL*){  type = 22;  }  potions.add(new Potion((int)(gc.getHitbox().x + gc.getHitbox().width / 2), (int)(gc.getHitbox().y - gc.getHitbox().height / 2), type));  return;  }  }  } } |

Cụ thể trong phương thức này sẽ có đầu vào là một Rectangle2D.Float đại diện cho hitboxattack của nhân vật. Trong phương thức ta sẽ tiến hành duyệt qua một arraylist là containers, sau đó kiểm tra xem là hộp chứa có được kích hoạt hay chưa, nếu được kích hoạt rồi thì ta kiểm tra tiếp đến xem có sự tương tác giữa hitbox của vật phẩm với hitboxattack của nhân vật hay không. Nếu có thì ta sẽ gán giá trị true cho thuộc tính doAnimation của thực thể. Sau đó thì ta sẽ khởi tạo một giá trị type có giá trị 21 (ở đây 21 chính là giá trị của lọ máu được đặt trong class constants). Chúng ta sẽ kiểm tra tiếp xem là kiểu thực thể của thùng mình đang kiểm tra là gì, nếu là BARREL thì ta sẽ gán cho giá trị type là 22 (ở đây 22 là giá trị của lọ năng lượng được đặt trong class constants). Và cuối cùng ta sẽ khởi tạo một lọ thuốc với tọa độ x là giá trị x của hitbox thùng chứa cộng thêm chiều dài của nó chia 2 và tọa độ y là giá trị y của hitbox thùng chứa cộng them chiều cao của nó chia 2 và type là kiểu lọ sẽ rơi ra. Điều này sẽ đảm bảo là lọ thuốc mới được rơi ra sau khi thùng chứa bị phá sẽ nằm chính giữa tại tọa độ ban đầu của thùng chứa.

### 4.4.3 Súng thần công

Súng thần công (Cannon) là vật thể xuất hiện cố định trên bản đồ tuy nhiên tay vì hỗ trợ nhân vật như các thực thể khác thì cannon lại có tác dụng tấn công nhân vật, gây ra một sự sát thương nhất định lên nhân vật Thực thể này sẽ đòi hỏi người chơi phải chọn ra cho mình đường đi phù hợp và khả năng né tránh các viên đạn.



Hình 4.19 Hình ảnh nhân vật đang né khỏi súng thần công

Cơ chế xác định mục tiêu để bắn của súng rất đơn giản. Thay vì để súng tự bắn với đường đạn cố định và bắn liên tục thì chúng ta có thể cài đặt để súng chỉ bắn mỗi khi có nhân vật di chuyển qua.

|  |
| --- |
| private void updateCannons(int[][] lvlData, Player player) {  for(Cannon c : cannons){  if(!c.doAnimation){  if(c.getTileY() == player.getTileY()){  if(isPlayerInRange(c,player)){  if(isPlayerInFrontOfCannon(c,player)){  if(*canCannonSeePlayer*(lvlData, player.getHitBox(), c.getHitbox(),c.getTileY())){  c.setAnimation(true);  }  }  }  }  }  c.update();  if(c.getAniIndex() == 4 && c.getAniTick() == 0){  shootCannon(c);  }  } } |

Cụ thể ta sẽ cho duyệt mảng cannons và bắt đầu xét, đầu tiên ta sẽ kiểm tra xem là cannon có đang ở trạng thái đứng yên hay không, nếu cannon ở trạng thái đứng yên thì ta sẽ bắt đầu kiểm tra tiếp xem người chơi có đang ở trong vùng bắn của cannon hay không.

Trong phương thức isPlayerInRange ta sẽ thiết lập một giá trị absValue được tính bằng trị tuyệt đối của hoành độ x của hitbox nhân vật trừ đi hoành độ x của hitbox cannon. Sau đó phương thức sẽ trả về xem liệu khoảng cách đó có nhỏ hơn hoặc bằng giá trị của 5 ô trên bản đồ hay không. Nếu có thì tức là nhân vật đang nằm trong khoảng có thể bắn của cannon. Tiếp đến ta sẽ kiểm tra xem nhân vật có đang ở đằng trước cannon hay không, ta sẽ xét xem hiện tại thì cannon là đang quay sang trái hay sang phải bằng cách kiểm tra kiểu của cannon, nếu cannon quay sang trái thì cũng ta sẽ kiểm tra hoành độ của hitbox cannon có lớn hơn hoành độ của nhân vật không nếu đúng thì tức là nhân vật đang nằm ở phía bên trái của cannon và giá trị trả về sẽ là true. Ngược lại nếu cannon đang quay sang phải và hoàng độ của hitbox cannon nhỏ hơn hoành độ của hitbox nhân vật thì tức là nhân vật nằm ở phía bên phải của cannon và cannon có thể bắn được, khi đó kết quả sẽ trả về là true nếu không đúng trường hợp nào thì phương thức sẽ trả về là false.  
Tiếp đến chúng ta sẽ kiểm tra xem liệu là cannon có thể nhìn thấy nhân vật hay không tức là giữa cannon và nhân vật không có gì chắn.

Đầu tiên chúng ta sẽ tính khai báo hai giá trị firstXTile và secondXTile bằng cách lấy giá trị hoành độ của hitbox first và second lần lượt chia cho kích thước của một ô trên bản đồ, cách tính như vậy là để xác định cột mà firstHitBox và secondHitBox thuộc về  
nếu giá trị của firstXTile lớn hơn secondXTile thì ta sẽ kiểm tra xem giữa hai ô đó có ô nào ở giữa mà là ô rắn không thể đi qua không bằng phương thứ isAllTilesClear với chỉ số xStart là sencondXTile và xEnd là firstXTile là y là yTile tức dòng mà nhân vật và cannon đang đứng, và việc gán firstXTile và secondXTile sẽ ngược lại nếu firstXTile không lớn hơn secondXTile.

Sau khi hoàn tất kiểm tra và cannon thỏa mãn hết các yêu cầu thì ta sẽ gán cho thuộc tính doAnimation của cannon là true rồi thoát khỏi vòng for. Tiếp đến ta sẽ kiểm tra xem giá trị của anindex có bằng 4 không và giá trị của anitick có bằng 0 không tức là cannon đã chạy hết 4 ảnh hay có thể hiểu là cannon đã thực hiện xong hành động hay chưa. Nếu thỏa mãn thì tức là cannon đã thực hiện xong rồi thì sẽ cho cannon bắn ra đạn bằng cách gọi đến phương thức shootCannon(c).

|  |
| --- |
| private void shootCannon(Cannon c) {  int dir = 1;  if(c.getObjType() == *CANNON\_LEFT*){  dir = -1;  }  projectiles.add(new Projectile((int)c.getHitbox().x, (int)c.getHitbox().y,dir)); } |

Và trong phương thức cannon chúng ta sẽ khai báo biến dir = 1 đại diện cho hướng của viên đạn (projectiles) quay sang bên phải. Ta kiểm tra xem nếu đây là cannon trái thì ta sẽ gán cho dir bằng -1 rồi thêm một projectile vào trong mảng projectile và sau đó viên đạn sẽ có thể được bắn ra.

### 4.4.4. Cạm bẫy

Đây là một vật thể mà chức năng chính là gài bẫy người chơi, khi nhân vật va chạm vào bẫy thì nhân vật sẽ chết nên đòi hỏi người chơi phải tỉnh táo và di chuyển khéo léo.



Hình 4.20 Hình ảnh cảm bẫy của trò chơi

# CHƯƠNG 5: CẢI THIỆN VÀ NÂNG CẤP

## 5.1. Vấn đề cơ sở dữ liệu

- Chưa thiết kế được cơ sở dữ liệu, chỉ đọc và lưu dữ liệu trên tệp

## 5.2. Một số tính năng có thể phát triển thêm

- Thêm nhiều quái hơn  
- Phát triển thêm map  
- Phát triển thêm các hiệu ứng của nhân vật  
- Phát triển cơ chế trò chơi

5.3. Cải thiện về quái  
- Quái thú có thể nhận diện được người chơi tốt hơn

- Quái thú có thể di chuyển và xuất hiện lại với thời gian chờ ngắn hơn

# PHÂN CHIA NHIỆM VỤ

|  |  |
| --- | --- |
| Mai Đức Vinh | Thiết kế những hình ảnh của trò chơi. Thiết lập cơ chế vòng lặp cho trò chơi, thiết lập nhân vật như hoạt ảnh di chuyển, nhảy, hiệu ứng đòn đánh, trọng lực ... Thiết lập cơ chế va chạm của người chơi với bản đồ. Thiết lập cơ chế chuyển bản đồ. Sáng tạo bản đồ cho trò chơi. Thiết lập màn hình dừng trò chơi. Thiết lập các nút bấm. |
| Trịnh Mạnh Quang | Làm vật thể của trò chơi như vật phẩm hỗ trợ, súng thần công, cạm bẫy ... Xử lý âm thanh cho trò chơi, giao diện tùy chọn, màn hình khi nhân vật hết máu. Sáng tạo bản đồ cho trò chơi. |
| Nguyễn Mạnh Hùng | Thiết lập các quái vật trong trò chơi như di chuyển, phát hiện nhân vật, tự sinh sản sau khi bị tiêu diệt ... Thiết lập những tương tác, các thông số cho quái vật và nhân vật. Sáng tạo bản đồ cho trò chơi. Thiết lập hiệu ứng tấn công của nhân vật. |

# TÀI LIỆU THAM KHẢO

Slide bài giảng của giảng viên Đào Thị Lệ Thủy – ĐH GTVT  
Tham khảo từ tài liệu của KaarinGaming: <https://github.com/KaarinGaming>

Tham khảo một số video bài giảng java của TITV: <https://www.youtube.com/@TITVvn>

Tham khảo cơ chế của trò chơi qua: <https://codelearn.io/sharing/co-ban-ve-game-loops-vong-lap-game>